IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Appellants:	Wing Lee	§	Group Art Unit: 3621
Serial No.:	10/619,296	§	Examiner: Winter, John M.
Filed:	July 14, 2003	§	Confirmation No.: 6314
For:	INTEGRATION INFRASTRUCTURE	§	

Mail Stop Appeal Brief-Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

APPEAL BRIEF

Dear Sirs:

This Appeal Brief is filed in support for the appeal in the above referenced application and is filed pursuant to the Notice of Appeal filed April 16, 2009. Appellant authorizes all required fees under 37 C.F.R. § 1.17 to be charged to Deposit Account No. 21-0765, of Sprint Communications Company L.P.

TABLE OF CONTENTS

I. REAL PARTY IN INTEREST 4

II. RELATED APPEALS AND INTERFERENCES 5

III. STATUS OF CLAIMS 6

IV. STATUS OF AMENDMENTS 7

V. SUMMARY OF CLAIMED SUBJECT MATTER 8

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL 26

VII. ARGUMENT 27

A. 35 U.S.C. § 101 Rejections - Claims 43-78. 30

1. Claims 43-78 were wrongly rejected because claims 43-78 are directed to statutory subject matter. 30

B. 35 U.S.C. § 103 Rejections - Claims 43-78. 32

1. Claims 43-78 were wrongly rejected because *Suarez* in view of *Hejlsberg* and *Bowman-Amuah* does not teach or suggest automatically publishing business events in accordance with the interactions between the front-office systems and the back-office systems. 32

a. Claims 43-78 were wrongly rejected because the *Final Office Action* improperly interpreted the term publish. 33

b. Claims 43-78 were wrongly rejected because *Bowman-Amuah* does not teach or suggest publishing business events in accordance with the interactions between the front-office systems and the back-office systems. 37

2. Claims 43-78 were wrongly rejected because *Suarez* in view of *Hejlsberg* and *Bowman-Amuah* does not provide any teaching or suggestion of a messaging system that automatically subscribes to the business events published by the enterprise integration layer. 39

3. Claims 43-78 were wrongly rejected because *Suarez* teaches away from the combination with Bowman-Amuah. 40

4. Claims 43-78 were wrongly rejected because the combination of *Suarez* with *Bowman-Amuah* would change the principle of operation of *Suarez*. 41

5. Claims 43-78 were wrongly rejected because *Suarez* in view of *Hejlsberg* and *Bowman-Amuah* does not teach or suggest defining an enterprise object model which defines objects that model the data and services provided by the back-office systems.42

6. Claims 43-78 were wrongly rejected because *Suarez* in view of *Hejlsberg* and *Bowman-Amuah* does not teach or suggest implementing, with a business object server of the enterprise integration layer coupled to the client access interfaces, data functions and service methods associated with the accessed objects that enable the interactions between the front-office systems and back-office systems. 43

7. Claims 43-78 were wrongly rejected because none of the limitations are optional, and conditional is not equivalent to optional. 44

C. 35 U.S.C. § 103 Rejections - Claims 56, 57, 59-61, 64, 66, 67, 69-71, and 73 46

1. Claims 56, 57, 59-61, 64, 66, 67, 69-71, and 73 were wrongly rejected because claim elements were not addressed by the *Final Office Action*. 46

VIII. CONCLUSION 49

IX. CLAIMS APPENDIX 50

X. EVIDENCE APPENDIX 69

XI. RELATED PROCEEDINGS APPENDIX 70

I. REAL PARTY IN INTEREST

The real party in interest in the present application is the following party: Sprint Communications Company L.P.

II. RELATED APPEALS AND INTERFERENCES

None.

III. STATUS OF CLAIMS

A. Total Number of Claims in the Application

The claims in the application are: 43-78

B. Status of All Claims in the Application

1. Claims canceled: 1-42
2. Claims withdrawn from consideration but not canceled: None
3. Claims pending: 43-78
4. Claims allowed: None
5. Claims rejected: 43-78

C. Claims on Appeal

The claims on appeal are: 43-78

IV. STATUS OF AMENDMENTS

No amendments were filed after the January 16, 2009 *Final Office Action*.

V. SUMMARY OF CLAIMED SUBJECT MATTER

The pending application discloses a more efficient manner for distributing data and events to multiple applications. *See, e.g.,* Application at 10, ¶ [0035] lines 1-2[1]. In a target state, an enterprise integration layer such as that described above would become aware of business events rather than merely passing on data and messages. *See, e.g.,* Application at 10, ¶ [0035] lines 2-3. This can be accomplished by integrating the enterprise integration layer with a business process integration engine that implements the core processes of the enterprise and feeds messages to the target systems. *See, e.g.,* Application at 10, ¶ [0035] lines 3-6. In this target state, back-office applications would typically need to be capable of generating or publishing business events. *See, e.g.,* Application at 10, ¶ [0035] lines 6-7. Middleware, such as the enterprise integration layer and the business process integration engine, would not necessarily be involved in business transactions. *See, e.g.,* Application at 10, ¶ [0035] lines 7-8. Applications that rely on batch loading would typically need to be capable of becoming online processes that are ready to consume events, that is, to subscribe to messages. *See, e.g.,* Application at 10, ¶ [0035] lines 9-10. Messaging-based data exchange would reduce the need for batch load processes but not eliminate it. *See, e.g.,* Application at 10, ¶ [0035] lines 10-11. Batch processes might continue to be the most effective means of providing data to some target applications. *See, e.g.,* Application at 10, ¶ [0035] lines 11-13. Messaging would provide an infrastructure to make events persistent, guarantee message

[1] 37 C.F.R. § 41.37 (c)(1)(v) provides that the "[s]ummary of claimed subject matter . . . shall refer to the specification by page and line number." The instant application was presented in numbered page and numbered paragraph form. As such, the citations to the specification support for the claimed subject matter will be presented in the following form: Application at ___ (page number), ¶ [____] (paragraph number), lines ____ (lines within the corresponding paragraph).

delivery, and transform messages for the target system. *See, e.g.,* Application at 10, ¶ [0035] lines 13-14. Messaging would also allow other applications interested in an event to easily plug in to the event. *See, e.g.,* Application at 10, ¶ [0035] lines 14-15. Data would be made available to applications in real-time, thus reducing or eliminating the need for replication and batch loads. *See, e.g.,* Application at 10, ¶ [0035] lines 15-17. Making the enterprise integration layer aware of business events may entail an assessment and realignment of an existing implementation of the layer and its tools, an alignment with the direction of future architecture, and an alignment with business process management and workflow strategy. *See, e.g.,* Application at 10, ¶ [0035] lines 17-20.

The enterprise integration layer is aware of business events rather than merely passing on data and messages. The modular architecture of the enterprise integration layer decouples the source and target systems, thereby providing enterprise-wide visibility to business events and transactions. The enterprise integration layer can define and generate business events and publish the business events to a message broker such that all applications that need to be aware of the events are made aware of the events. Business events are key milestones within a process flow. The definition and automatic generation of business events by the enterprise integration layer can make an enterprise business event-aware. When key business events are available, applications can readily use them rather than having to deduce them by replicating data and applying rules, which can be time-consuming and error prone.

On the occasion when the pertinent paragraph is contained on multiple pages, the paragraph line numbering will begin anew on subsequent pages.

The enterprise integration layer may store event notification rules that define criteria for determining when to publish what information to the message broker. For example, an event can be considered to occur upon the creation, reading, updating, or deleting of an object, upon the invocation of a particular method, upon the evaluation of an expression, or when similar activities occur. Rules regarding such functions can be associated with particular interactions or transactions that are brokered between the front-office and back-office systems by the enterprise integration layer.

When the event notification rules of the enterprise integration layer indicate that components elsewhere in the enterprise need to be aware of the event, then the event is published to the message broker. For example, the enterprise integration layer publishes events based on the transactions it is brokering, such as data creation, updating, and deletion transactions. The message broker may exchange messages with other systems using the publish/subscribe paradigm. The message broker may automatically subscribe to all of the business events published by the enterprise integration layer. The message broker may then make the entire enterprise aware of the business events by publishing messages in a common format on a message bus or a message queue. For example, the enterprise integration layer, one or more of the front-office systems, and/or one or more of the back-office systems can subscribe to information published by the message broker. Therefore, the message broker provides a foundation for business process automation by allowing integration of both data and business logic across applications.

A detailed discussion of components of the architecture of the enterprise integration layer and the message broker follow. The enterprise integration layer may include an enterprise object

model that defines objects that represent the data and services provided by the back-office systems. (See, *e.g.*, Application at 14, ¶ [0043], lines 1-12 and ¶ [0044], lines 1-13). For example, the enterprise object model may be defined through the use of a Unified Modeling Language (UML) graphical editor.

Front-office applications may access the objects in the enterprise object model through standardized client access interfaces that enable standardized access to back-office data and services through a plurality of different technologies. (See, *e.g.*, Application at 12, ¶ [0040], lines 1-12, at 13, ¶ [0040], lines 1-11, and at 17, ¶ [0051], lines 1-12). By providing access to objects that model the back-office data and services, the client access interfaces eliminate the tight coupling between the front-office systems and the back-office systems. Further, the modular architecture provided by the enterprise object model allows rapid changes to support new requirements.

Upon an object in the enterprise object model being accessed through the client access interfaces, a business object server may implement any data functions and service methods associated with the accessed object. The business object server may implement the data functions and service methods by performing object assembly, object disassembly, and service invocation functions. (See, *e.g.*, Application at 16, ¶ [0048], lines 1–17, ¶ [0050], line 11).

A set of adaptors may be used to map between the data services modeled in the enterprise object model and the data and functions of the back-office systems. (See, *e.g.*, Application at 14, ¶ [0044], lines 1–16, ¶[0046], line 2). The enterprise integration layer also includes a rules engine that defines and stores rules regarding criteria for when to publish the business events and rules regarding transforming data from a common format, such as the format of the enterprise

object model, to a format of the back-office systems. (See, *e.g.*, Application at 14, ¶ [0045], lines 1-6 and at 15, ¶ [0045], lines 1-14).

The enterprise integration layer further includes a business event repository that contains definitions of the business events that are of interest to a plurality of the computing applications and also identifies all of the publishers for each of the business events. (See, *e.g.*, Application at 15, ¶ [0046], lines 1–16, ¶ [0046], line 2). Events are a key milestone within a process flow. (See, *e.g.*, Application at 13, ¶ [[0041], lines 1-15). Upon the occurrence of an event, if the business event repository indicates that components elsewhere in the enterprise need to be aware of the event, then the event is published to a message broker. (See, *e.g.*, Application at 17, ¶ [0050], lines 1-11).

The message broker is a middleware layer that can connect disparate applications and transport information in a consistent format between systems. Events may be published to a message bus or a message queue on the message broker in a standard format to minimize or eliminate data replication. Also, the message broker may include connectors and adapters for integrating applications with the messaging environment. A source application may publish an event to the message broker through a source application adapter. The source application adaptor transforms data of the business event from a format of the source application to a standard data format. A target application adaptor may subscribe to events desired by a target application, transform data of the event from the standard format to a format of the target application so that the target application can retrieve a message. The middleware architecture of the message broker can eliminate the need for tight integration between application programming

interfaces and therefore provide a more flexible environment. (See, *e.g.*, Application at 20, ¶ [0062], lines 1–22, ¶ [0065], line 16).

Claim 43 is independent and is directed to a method for making computing applications throughout an enterprise aware of business events, comprising: defining objects in an enterprise object model that model data and services provided by back-office systems, *See, e.g.,* Application at 21, ¶ [0062] lines 14-19; brokering interactions, by an enterprise integration layer, between the back office systems that provide data and services and front-office systems that use the enterprise integration layer to access the data and the services provided by the back office-systems through the interactions, *See, e.g.,* Application at 12, ¶ [0039] line 4 - ¶ [0040] line 6, brokering the interactions comprising: receiving, from the front-office systems, accesses to objects of the enterprise object model in the enterprise integration layer through client access interfaces of the enterprise integration layer, wherein each of the client access interfaces corresponds with a different technology and provides a standardized interface through which the front-office systems access the objects of the enterprise object model, *See, e.g.,* Application at 12, ¶ [0040] lines 12–13, ¶ [0040] line 3; implementing, with a business object server of the enterprise integration layer coupled to the client access interfaces, data functions and service methods associated with the accessed objects that enable the interactions between the front-office systems and back-office systems, *See, e.g.,* Application at 16, ¶ [0048] lines 4–7; and transforming, with a set of adapters of the enterprise integration layer coupled to the business object server, the accessed objects into a format of the back-office systems corresponding with the implementation of the data functions and the service methods associated with the accessed objects, *See, e.g.,* Application at 15, ¶ [0046] line 16, ¶ [0046] line 2; automatically publishing,

by the enterprise integration layer, business events in accordance with the interactions between the front-office systems and back-office systems, *See, e.g.,* Application at 13, ¶ [0041] lines 1–15; automatically subscribing, by a messaging system coupled to the enterprise integration layer, to the business events published by the enterprise integration layer, *See, e.g.,* Application at 13, ¶ [0041] lines 12-15 and Application at 16, ¶ [0049] lines 1-7; and automatically generating, by the messaging system, for each of the subscribed business events a message that makes computing applications that are interested in the business event aware of the business event, *See, e.g.,* Application at 13, ¶ [0041] lines 7-12.

Claim 44 depends on claim 43 and adds the limitation of defining and storing rules in a rules engine within the enterprise integration layer, the rules including rules regarding when to automatically publish the business events in accordance with the interactions, rules regarding the transforming of the accessed objects of the enterprise object model to the format of the back-office systems, and rules regarding mapping each of the back-office systems to an appropriate adaptor in the set of adaptors, wherein the business events are automatically published in accordance with the interactions and the rules regarding when to automatically publish the business events.

Claim 45 depends on claim 43 and adds the limitations of defining, in a business event repository within the enterprise integration layer, the business events that are of interest to the computing applications; and identifying, in the business event repository, all of the publishers for each of the business events.

Claim 46 depends on claim 43 and adds the limitation of holding, in a metadata repository within the enterprise integration layer, metadata supplied by the set of adaptors that

enables the transforming of the accessed objects of the enterprise object model to the format of the back-office systems.

Claim 47 depends on claim 43 and adds the limitation that the business events are key milestones within a process flow.

Claim 48 depends on claim 43 and adds the limitation that implementing data functions and service methods associated with the accessed objects further comprises: performing one or more of object assembly, object disassembly, and service invocation functions, wherein performing object assembly includes creating a composite object by aggregating data from a plurality of the back-office systems, performing object disassembly includes breaking a composite object into multiple objects for storage in at least one of the back-office systems, and performing service invocation includes determining which functions to invoke on one or more of the back-office systems.

Claim 49 depends on claim 43 and adds the limitation that one of the business events occurs upon the implementation of the data functions and the service methods associated with the access objects, including one or more of creating data, reading data, updating data, deleting data, and invoking one of the service methods.

Claim 50 depends on claim 43 and adds the limitation that the automatic generation of the message for each subscribed business event further comprises: mapping, by one or more adaptors of a transformation layer of the messaging system, data corresponding to the business events published by the enterprise integration layer between a format of a source of the business events and a format of the computing applications.

Claim 51 depends on claim 50 and adds the limitations of transforming, by a source application adaptor of the one or more adaptors, data related to a business event from a format of

a source of the business event to a standard data format; and transforming, by a target application adaptor of the one or more adaptors, data from the standard data format to a format of a target subscribed to the business event.

Claim 52 is independent and is directed to a method for making computing applications throughout an enterprise aware of business events, comprising: brokering interactions, by an enterprise integration layer, between back office systems that provide data and services and front-office systems that use the enterprise integration layer to access the data and the services provided by the back office-systems through the interactions, *See, e.g.,* Application at 12, ¶ [0039] line 4 - ¶ [0040] line 6, brokering the interactions comprising: receiving, from the front-office systems, accesses to common format descriptions in the enterprise integration layer of the data and the services provided by the back-office systems through client access interfaces of the enterprise integration layer, wherein each of the client access interfaces correspond with a different technology and provide a standardized interface through which the front-office systems access the common format descriptions of the data and the services provided by the back-office systems, *See, e.g.*, Application at 12, ¶ [0040] lines 12–13, ¶ [0040] line 3; implementing, with a business object server of the enterprise integration layer coupled to the client access interfaces, data functions and service methods associated with the accessed common format descriptions of the data and the services, wherein implementing the data functions and the service methods enable the interactions between the front-office systems and back-office systems *See, e.g.*, Application at 16, ¶ [0048] lines 4-7; and transforming, with a set of adapters of the enterprise integration layer coupled to the business object server, the accessed common format descriptions of the data and the services into a format of the back-office systems corresponding with the

implementation of the data functions and the service methods associated with the accessed common format descriptions of the data and the services *See, e.g.*, Application at 15, ¶ [0046] line 16, ¶ [0046] line 2; defining and storing rules in a rules engine within the enterprise integration layer, the rules including rules regarding when to publish business events in accordance with the interactions between the front-office systems and the back-office systems, *See e.g.*, Application at 14, ¶ [0045] lines 1–15, ¶ [0045] line 14; automatically publishing, by the enterprise integration layer, the business events in accordance with the rules, *See, e.g.*; Application at 13, ¶ [0041] lines 1-15; automatically subscribing, by a messaging system coupled to the enterprise integration layer, to the business events published by the enterprise integration layer, *Se, e.g.*, Application at 13, ¶ [0041] lines 12-15 and Application at 16, ¶ [0049] lines 1-7; and automatically generating, by the messaging system, for each of the subscribed business events a message that makes computing applications that are interested in the business event aware of the business event, *See, e.g.*, Application at 13, ¶ [0041] lines 7-12.

Claim 53 depends on claim 52 and adds the limitation of defining objects in an enterprise object model that model the data and the services provided by the back-office systems, wherein the objects in the enterprise object model are the common format descriptions of the data and the services provided by the back-office systems.

Claim 54 depends on claim 52 and adds the limitation of defining, in a business event repository within the enterprise integration layer, the business events that are of interest to the computing applications.

Claim 55 depends on claim 52 and adds the limitation of holding, in a metadata repository within the enterprise integration layer, metadata supplied by the set of adaptors that

enables the transforming of the accessed common format descriptions of the data and the services to the format of the back-office systems.

Claim 56 depends on claim 52 and adds the limitation of providing distributed transactional quality of service through a transaction processor within the enterprise integration layer.

Claim 57 depends on claim 52 and adds the limitation of making data persistent within a local data store of the enterprise integration layer.

Claim 58 depends on claim 52 and adds the limitation that the business events are key milestones within a process flow.

Claim 59 depends on claim 52 and adds the limitation of using previously existing infrastructure services within the enterprise for the enterprise integration layer.

Claim 60 depends on claim 59 and adds the limitation that the previously existing infrastructure services are selected from a group of services consisting of a naming and directory service, a security service, and an application management and monitoring system.

Claim 61 depends on claim 60 and adds the limitation that the previously existing infrastructure services include each of a group of services comprising a naming and directory service, a security service, and an application management and monitoring system.

Claim 62 is independent and is directed to a method for making computing applications throughout an enterprise aware of business events, comprising: brokering interactions, by an enterprise integration layer, between back office systems that provide data and services and front-office systems that use the enterprise integration layer to access the data and the services provided by the back office-systems through the interactions, *See, e.g.*, Application at 12, ¶ [0039] line 4 - ¶ [0040] line 6, brokering the interactions comprising: receiving, from the front-

office systems, accesses to common format descriptions in the enterprise integration layer of the data and the services provided by the back-office systems through client access interfaces of the enterprise integration layer, wherein each of the client access interfaces corresponds with a different technology and provides a standardized interface through which the front-office systems access the common format descriptions of the data and the services provided by the back-office systems, *See, e.g.*, Application at 12, ¶ [0040] lines 12–13, ¶ [0040] line 3; implementing, with a business object server of the enterprise integration layer coupled to the client access interfaces, data functions and service methods associated with the accessed common format descriptions of the data and the services, wherein implementing the data functions and the service methods enable the interactions between the front-office systems and back-office systems; and transforming, with a set of adapters of the enterprise integration layer coupled to the business object server, the accessed common format descriptions of the data and the services into a format of the back-office systems corresponding with the implementation of the data functions and the service methods associated with the accessed common format descriptions of the data and the services, *See, e.g.*, Application at 16, ¶ [0048] line 4-7 and at 22, ¶ [0065] lines 1-16; defining, in a business event repository within the enterprise integration layer, business events that are of interest to computing applications in the enterprise; identifying, in the business event repository, all publishers for each of the business events, *See, e.g.*, Application at 13, ¶ [0042] lines 1-5 and at 15, ¶ [0046] lines 1-6; automatically publishing, by the enterprise integration layer, the business events in accordance with the interactions between the front-office systems and back-office systems, *See, e.g.*, Application at 13, ¶ [0041] lines 1-15; automatically subscribing, by a messaging system coupled to the enterprise integration layer,

to the business events published by the enterprise integration layer, *See, e.g.*, Application at 13, ¶ [0041] lines 12-15 and at 16, ¶ [0049] lines 1-7; and automatically generating, by the messaging system, for each of the subscribed business events a message that makes the computing applications that are interested in the business event aware of the business event, *See, e.g.*, Application at 13, ¶ [0041] lines 7-12.

Claim 63 is dependant on claim 62 and adds the limitation of defining objects in an enterprise object model that model the data and the services provided by the back-office systems, wherein the objects in the enterprise object model are the common format descriptions of the data and the services provided by the back-office systems.

Claim 64 is dependant on claim 62 and adds the limitation of defining and storing rules in a rules engine within the enterprise integration layer, the rules including rules regarding when to automatically publish the business events in accordance with the interactions, rules regarding the transforming of the accessed common format descriptions of the data and the services into the format of the back-office systems, and rules regarding mapping each of the back-office systems to an appropriate adaptor in the set of adaptors, wherein the business events are automatically published in accordance with the interactions and the rules regarding when to automatically publish the business events.

Claim 65 is dependant on claim 62 and adds the limitation of holding, in a metadata repository within the enterprise integration layer, metadata supplied by the set of adaptors that enables the transforming of the accessed common format descriptions of the data and the services to the format of the back-office systems.

Claim 66 is dependant on claim 62 and adds the limitation of providing distributed transactional quality of service through a transaction processor within the enterprise integration layer.

Claim 67 is dependant on claim 62 and adds the limitation of making data persistent within a local data store of the enterprise integration layer.

Claim 68 is dependant on claim 62 and adds the limitation that the business events are key milestones within a process flow.

Claim 69 is dependant on claim 62 and adds the limitation of using previously existing infrastructure services within the enterprise for the enterprise integration layer.

Claim 70 is dependant on claim 69 and adds the limitation that the previously existing infrastructure services are selected from a group of services consisting of a naming and directory service, a security service, and an application management and monitoring system.

Claim 71 is dependant on claim 70 and adds the limitation that the previously existing infrastructure services include each of a group of services comprising a naming and directory service, a security service, and an application management and monitoring system.

Claim 72 is independent and is directed to a method for making computing applications throughout an enterprise aware of business events, comprising: defining objects in an enterprise object model that model data and services provided by back-office systems, *See, e.g.*, Application at 18, ¶ [0056] lines 1-6; brokering interactions, by an enterprise integration layer, between the back office systems that provide data and services and front-office systems that use the enterprise integration layer to access the data and the services provided by the back office-systems through the interactions, brokering the interactions comprising: receiving, from the

front-office systems, accesses to objects of the enterprise object model in the enterprise integration layer through client access interfaces of the enterprise integration layer, wherein each of the client access interfaces corresponds with a different technology and provides a standardized interface through which the front-office systems access the objects of the enterprise object model, *See, e.g.*, Application at 12, ¶ [0040] lines 12–13, ¶ [0040] line 3; implementing, with a business object server of the enterprise integration layer coupled to the client access interfaces, data functions and service methods associated with the accessed objects that enable the interactions between the front-office systems and back-office systems, *See, e.g.*, Application at 16, ¶ [0048] lines 4-7; and transforming, with a set of adapters of the enterprise integration layer coupled to the business object server, the accessed objects into a format of the back-office systems corresponding with the implementation of the data functions and the service methods associated with the accessed objects, *See, e.g.*, Application at 15, ¶ [0046] line 16 -, ¶ [0047] line 2; defining and storing rules in a rules engine within the enterprise integration layer, the rules including rules regarding when to publish business events in accordance with the interactions between the front-office systems and the back-office systems; automatically publishing, by the enterprise integration layer, the business events in accordance with the rules, *See, e.g.*, Application at 14, ¶ [0045] lines 1–15, ¶ [0045] line 14; automatically subscribing, by a messaging system coupled to the enterprise integration layer, to the business events published by the enterprise integration layer, *See, e.g.*, Application at 13, ¶ [0041] lines 12-15 and at 16, ¶ [0049] lines 1-7; transforming, by a source adaptor of the messaging system, data related to at least one of the business events from a format of a source of the at least one of the business events to a common data format, *See, e.g.*, Application at 20, ¶ [0062] lines 1–21, ¶ [0062] line

19; publishing, by a message interface of the messaging system, the at least one of the business events and the transformed data related to the at least one of the business events in the common data format, *See, e.g.*, Application at 13, ¶ [0041] lines 1-15 and at 22, ¶ [0065] lines 1-16; transforming, by a target application adaptor of the messaging system, the data related to the at least one of the business events from the common data format to a format of a target application subscribed to the at least one of the business events published by the message interface of the messaging system, *See, e.g.*, Application at 22, ¶ [0065] lines 12-16; and processing the at least one of the business events by the target application, *See, e.g.*, Application at 22, ¶ [0067] lines 1–23, ¶ [0067] line 11.

Claim 73 is dependant on claim 72 and adds the limitation that the at least one of the business events and the transformed data related to the at least one of the business events are combined in a single packet and published by the messaging interface of the messaging system or are independently published by the messaging interface of the messaging system.

Claim 74 is dependant on claim 72 and adds the limitation that the business events are key milestones within a process flow.

Claim 75 is independent and is directed to a method for making computing applications throughout an enterprise aware of business events, comprising: defining objects in an enterprise object model that model data and services provided by back-office systems, *See, e.g.*, Application at 18, ¶ [0056] lines 1-6; brokering interactions, by an enterprise integration layer, between the back office systems that provide data and services and front-office systems that use the enterprise integration layer to access the data and the services provided by the back office-systems through the interactions, brokering the interactions comprising: receiving, from the

front-office systems, accesses to objects of the enterprise object model in the enterprise integration layer through client access interfaces of the enterprise integration layer, wherein each of the client access interfaces corresponds with a different technology and provides a standardized interface through which the front-office systems access the objects of the enterprise object model, *See, e.g.*, Application at 12, ¶ [0040] lines 12–13, ¶ [0040] line 3; implementing, with a business object server of the enterprise integration layer coupled to the client access interfaces, data functions and service methods associated with the accessed objects that enable the interactions between the front-office systems and back-office systems, *See, e.g.*, Application at 16, ¶ [0048] lines 4-7; and transforming, with a set of adapters of the enterprise integration layer coupled to the business object server, the accessed objects into a format of the back-office systems corresponding with the implementation of the data functions and the service methods associated with the accessed objects, *See, e.g.*, Application at 15, ¶ [0046] lines16 – ¶ [0047] line 2; defining, in a business event repository within the enterprise integration layer, business events that are of interest to computing applications in the enterprise; identifying, in the business event repository, all publishers for each of the business events, *See, e.g.*, Application at 13, ¶ [0042] lines 1-5 and at 15, ¶ [0046] lines 1-6; defining and storing rules in a rules engine within the enterprise integration layer, the rules including rules regarding when to publish the business events in accordance with the interactions between the front-office systems and the back-office systems, *See, e.g.*, Application at 14, ¶ [0045] lines 1–15, ¶ [0045] line 14; automatically publishing, by the enterprise integration layer, the business events in accordance with the rules, *See, e.g.*, Application at 14, ¶ [0045] lines 1–15, ¶ [0045] line 14; automatically subscribing, by a messaging system coupled to the enterprise integration layer, to the business events published by

the enterprise integration layer, *See, e.g.*, Application at 13, ¶ [0041] lines 12-15 and at 16, ¶ [0049] lines 1-7; automatically generating, by the messaging system, for each of the subscribed business events a message that makes the computing applications that are interested in the business event aware of the business event, *See, e.g.*, Application at 20, ¶ [0062] lines 1–21, ¶ [0062] line 19.

Claim 76 is dependant on claim 75 and adds the limitation that the automatic generation of the message for each subscribed business event further comprises: mapping, by one or more adaptors of a transformation layer of the messaging system, data corresponding to the business events published by the enterprise integration layer between a format of a source of the business events and a format of the computing applications.

Claim 77 is dependant on claim 76 and adds the limitations of transforming, by a source application adaptor of the one or more adaptors, data related to a business event from a format of a source of the business event to a standard data format; and transforming, by a target application adaptor of the one or more adaptors, data from the standard data format to a format of a target subscribed to the business event.

Claim 78 is dependant on claim 75 and adds the limitation that the business events are key milestones within a process flow.

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

1. Whether claims 43-78 are directed to statutory subject matter under 35 U.S.C. § 101.

2. Whether claims 43-78 are rendered obvious under 35 U.S.C. § 103(a) by *Suarez,* U.S. Pat. No. 5,790,789, (hereinafter *Suarez*) in view of *Hejlsberg*, et al., U.S. Pat. No. 7,165,239 (hereinafter *Hejlsberg*), and further in view of Bowman-Amuah, U.S. Pat. No. 6,742,015 (hereinafter *Bowman-Amuah*).

VII. ARGUMENT

Applicant respectfully submits that none of the applied art alone or in combination teaches or suggests an enterprise integration layer and a messaging system as described by the pending application and claimed. All of the pending rejections rely on *Suarez* as a base reference. *Suarez* is directed to a peer-to-peer-type distributed computer environment that relies on agents to provide interoperability for point-to-point communications between objects. In contrast, the pending claims are directed in part to a client-server-type distributed computer environment that relies on an enterprise integration layer to integrate front-office systems with back-office systems as well as a messaging broker that uses a publish/subscribe methodology that allows various applications to become aware of business events without requiring back-office systems to communicate directly with each front-office system requiring knowledge of a business event. The decoupling of interactions between the front-office systems and the back-office systems through the enterprise integration layer in conjunction with decoupling of messaging between front-office systems and back-office systems through the messaging broker promotes ease of modification to the system since existing back-office applications do not need to be reprogrammed each time a new front-office application is added and vice versa.

By utilizing a distributed computing architecture where front-office systems interact through the centralized enterprise integration layer to use back-office systems, the enterprise integration layer of the pending claims may become business event-aware and may publish business events based on the interactions between the front-office systems and the back-office systems through the enterprise integration layer. *Suarez* is directed to a wholly different paradigm in distributed computing. The fundamental differences in the distributed computing

paradigms used by *Suarez* and the pending claims highlight the differences discussed in detail below.

At the highest level, the distributed computing architecture of *Suarez* does not have any central means for providing interoperability between services and thus does not have a central entity that may become business event-aware and may publish business events based on interactions between front-office systems and back-office systems, like the claimed enterprise integration layer. Rather, *Suarez* relies on agents and bus agents distributed at each host computer to provide interoperability with other agents and services. While *Suarez* may disclose the use of bus agents, the bus agents of *Suarez* simply provide administrative oversight of multiple agents. For example, the bus agents of *Suarez* may route messages to the appropriate agent, monitor the health of agents and detect failures, manipulate message flow, and manipulate the content of the messages similar to other agents.

To further illustrate this distinction between a peer-to-peer-type distributed computing environment of *Suarez* and a three-tier client-server-type distributed computing environment as claimed, Applicant respectfully notes Fig. 5 of *Suarez* depicting the service-to-service communication flow through *Suarez* 's distributed computing environment using agents and bus agents. As shown in Fig. 5 of *Suarez,* all of the communications are point-to-point communications. That is one service communicates with one agent, which in turn communicates with one bus agent, which communicates with another bus agent, which communicates with another agent, which communicates with the destination service. Point-to-point communications is **not** publishing.

In contrast, Fig. 3 of the pending application depicts a many-to-one relationship between front-office applications 348 and the service broker 300 (i.e., the enterprise integration layer) as well as a one-to-many relationship between the service broker 300 and the back-office systems 350. Further, as discussed in paragraphs 0048-0050 of the pending application, the business object server 320 of the service broker 300 may support object assembly which aggregates data attributes from multiple back-office systems 350 into a composite object. Similarly, the business object server 320 may support object disassembly which breaks a composite object into multiple objects for storage in back-office systems 350. Further, the business object server 320 may support composite business services such that a single method call defined in the enterprise object model may translate into multiple service invocations. Therefore, an exemplary communication flow using the disclosed and claimed enterprise integration layer may include multiple front-office applications communicating with the enterprise integration layer to access objects of the enterprise object model, and the service broker implementing data functions or service methods associated with each of the access objects on multiple back-office systems (i.e., many-to-one and one-to-many).

Furthermore, *Suarez* does not teach or suggest a system in which interactions between front-office systems and back-office systems occur though an enterprise integration layer **and** messaging between front-office systems and back-office systems occur through a messaging broker. As touched on above, decoupling both the interactions and the messaging between front-office systems and back-office systems promotes ease of modification to the system since existing back-office applications do not need to be reprogrammed each time a new front-office application is added and vice versa. However, *Suarez* at most, might teach a message broker that

decouples messaging between services only if the agents of *Suarez* are equated with the message broker of the pending claims (which is clearly not the case), but does not teach or suggest a centralized enterprise integration layer that decouples interactions between services. In other words, the distributed computing architecture described by the pending application and claimed includes two centralized middleware entities, an enterprise integration layer and a messaging broker, whereas *Suarez* only discloses distributed middleware entities on each computer, agents.

Accordingly, Applicant respectfully submits that *Suarez* and the other applied art alone or in combination do not teach or suggest an enterprise integration layer and a messaging system as claimed in claims 43-78.

A. 35 U.S.C. § 101 Rejections - Claims 43-78.

1. Claims 43-78 were wrongly rejected because claims 43-78 are directed to statutory subject matter.

The *Final Office Action* dated January 16, 2009 (henceforth "*Final Office Action*") rejected claims 43-78 "under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter." Claim 43 is directed to a "method for making computing applications throughout an enterprise aware of business events" and requires "an enterprise integration layer," "a business object server", and "a messaging system." The CAFC held that a method claim that is tied to a particular machine is statutory under § 101. *In re Bilski*, 545 F.3d 943, 961 (Fed. Cir. 2008). The CCPA, whose opinions like those of the CAFC are binding on the Patent Office, held that a claimed method that was performed by a digital computer was statutory. *Application of Bernhart*, 417 F.2d 1395, 1398-1401 (CCPA 1969) (holding that the apparatus and method claims were directed to statutory subject matter). The *Bernhart* decision also states that a digital

computer, programmed to perform a novel and unobvious function is, in fact, a new machine, or at least an improvement to an old machine, and thus statutory. Id. at 1400. Specifically, "if a machine is programmed in a certain new and unobvious way, it is physically different from the machine without that program; its memory elements are differently arranged." Id. at 1400. The Federal Circuit referred to *Bernhart* favorably and held that a general purpose computer, or microprocessor, programmed to carry out an algorithm becomes a special purpose computer and thus a new machine. *WMS Gaming, Inc. v. International Game Technology*, 184 F.3d 1339, 1348 (Fed. Cir. 1999) ("A general purpose computer, or microprocessor, programmed to carry out an algorithm creates "a new machine, because a general purpose computer in effect becomes a special purpose computer once it is programmed to perform particular functions pursuant to instructions from program software."). Accordingly, under the binding precedent of *Bernhart* and *WMS Gaming*, a computer that is programmed to perform a novel and unobvious method is, by definition, a particular machine for purpose of the § 101 analysis under *Bilski*. See also *Ex Parte Wang*, 2008 WL 4448241,*7, n.3 (Bd.Pat.App. & Interf. Sept. 30, 2008) (citing approvingly *Application of Bernhart*).

Claim 43 requires "an enterprise integration layer," "a business object server", and "a messaging system" all of which are described in detail in the specification as filed. Such components comprise a particular machine per the *Bilski* and *Bernhart* decisions. Additionally, claim 43 does not merely recite steps that can be performed by a human, a factor that militated in favor of a finding in *Bernhart* that the method claim was statutory. For at least these reasons, claim 43 and its dependent claims are directed to statutory subject matter.

Similarly, independent claims 52, 62, 72, and 75 each require "an enterprise integration layer," "a messaging system," "a business object server," and "a set of adapters." Such components comprise a particular machine per the *Bilski* and *Bernhart* decisions. Additionally, each of claims 52, 62, 72, and 75 do not merely recite steps that can be performed by a human, a factor that militated in favor of a finding in *Bernhart* that the method claim was statutory. For at least these reasons, each of claims 52, 62, 72, and 75 and their respective dependent claims are directed to statutory subject matter.

B. 35 U.S.C. § 103(a) Rejections - Claims 43-78.

1. Claims 43-78 were wrongly rejected because *Suarez* in view of *Hejlsberg* and *Bowman-Amuah* does not teach or suggest automatically publishing business events in accordance with the interactions between the front-office systems and the back-office systems.

As noted by the United States Supreme Court in *Graham v. John Deere Co. of Kansas City*, an obviousness determination begins with a finding that **"the prior art as a whole in one form or another contains all" of the elements of the claimed invention**. *See Graham v. John Deere Co. of Kansas City*, 383 U.S. 1, 22 (U.S. 1966).

Claim 43 recites, "automatically publishing, by the enterprise integration layer, business events in accordance with the interactions between the front-office systems and back-office systems." In addressing this feature, the *Final Office Action* makes two contradictory statements. In the response to the Applicant's arguments that begin on page 2 of the *Final Office Action*, the *Final Office Action* states that "publishing" is taught by *Suarez* using an incorrect interpretation

of the term "publishing" and then later, on page 6, admits that *Suarez* does not disclose publishing, but erroneously asserts that *Bowman-Amuah* teaches "publishing."

a. Claims 43-78 were wrongly rejected because the *Final Office Action* improperly interpreted the term "publish."

In the Response to Arguments section of the *Final Office Action* on pages 2-3, the *Final Office Action* states that *Suarez* discloses "publish." The *Final Office Action* further states that the:

> term "publish" is construed "to make generally known", there is no limitation in the claimed invention as to what parties the information is made know [sic] to (i.e. the difference between a broadcasting of information and a sending of information to a specific destination), therefore the Examiner submits that the process as described by *Suarez* "As seen in FIG. 2, the illustrated process flow 22 is typically comprised of one or more activities 24 which are performed on a work item 25 in a prescribed sequence. It is through the communication of defined services 16 with one another that the various processes and process flows 22 are performed and tasks are accomplished. Associated with each process or process flow 22 is a variety of other information referred to as attachments" meets the limitation of the claimed invention. Information is transmitted by a first party and received by a second, which is construed by the Examiner as an act of "publishing".

Construing information transmitted by a first party and received by a second party as "publishing" is inconsistent with the definition provided by the *Final Office Action* that states that "publish" is "to make generally known." "To make generally known" is to make something public, i.e., known to more than one. Transmitting by a first party and received by a second party is a point-to-point communication and the electronic equivalent to sending someone a letter – an act that is not considered to be publishing.

Furthermore, in construing the meaning of a claim limitation, it is entirely proper to look to the specification in order to interpret what the inventor intended by the claim term. *In re Sneed*, 710 F2d. 1544, 1548, 218 U.S.P.Q. 385, 388 (Fed. Cir. 1983) ("It is axiomatic that, in proceedings before the PTO, claims in an application are to be given their broadest reasonable interpretation consistent with the specification, . . . , and that claim language should be read in light of the specification as it would be interpreted by one of ordinary skill in the art."); *In re Marosi*, 710 F.2d 799, 802-03, 218 U.S.P.Q. 289, 292 (Fed. Cir. 1983) ("It is well established that 'claims are not to be read in a vacuum, and limitations therein are to be read in light of the specification . . .'"); *In re Ehrreich*, 590 F.2d 902, 907, 200 U.S.P.Q. 504, 508 (CCPA 1979). In reading the claims in light of the specification of the pending application, it becomes clear that the interpretation of information transmitted by a first party and received by a second as an act of "publishing" is inconsistent with the usage of the term "publishing" in the pending application which contrasts "publishing" with "point-to-point" communication. Consider, for example, paragraphs [0006] and [0007] from the pending application which are reproduced below for convenience and state:

> [0006] Several types of topology can support messaging. These include publish/subscribe, point-to-point, hub and spoke, bus, and distributed hub. Publish/subscribe messaging is organized around topics. A publisher sends a message to a topic and any interested subscriber can receive the message from the topic. Publish/subscribe messaging is typically used when multiple subscribers might be interested in the same message. It is appropriate for notification messages for which no response is required upon consumption. It is also useful for enterprise-level messages such as account creation, account termination, and subscription suspension. For example, a message server could publish an "account created" event after an account has been created and subscribers could consume the message.

> [0007] Point-to-point messaging is based on message queues. A producer sends a message to a specified queue and a consumer receives messages from the queue. Multiple senders and receivers are possible for a queue but an individual message can be delivered to only one receiver. Point-to-point messaging is typically used when only one consumer exists and the message is targeted for a known application. It is also useful when successful consumption by the target system is a requirement since messages stay in the queue until the receiver picks them up. As an example, point-to-point messaging would be appropriate within a telecommunications company when a message to reserve a mobile telephone number is transmitted. Such a message would typically be transmitted to only one consumer.

The specification of the pending application in paragraphs [0040] and [0041] also further clarifies that "publishing" is not equivalent to "point-to-point" communication and states:

> . . . Service Broker uses back-office metadata repositories that can contain business rules, data transformation rules, and rules for publishing events via Message Broker. This rule-driven architecture also supports validation rules, data and service access rules, caching rules, and event generation rules. Service Broker uses a component-based architecture based on an enterprise-wide object model that represents all major business entities in the operations domain. This modular architecture allows rapid changes to support new requirements. Service Broker provides a transformation mechanism for mapping between an enterprise object model format and back-office system formats. It uses commercially available transformation engines (such as XSLT) rather than hard coding.
>
> [0041] Service Broker can define and generate business events and publish the business events to all applications that need to be aware of the events. Business events are key milestones within a process flow such as the completion of a credit check or the creation of an account. Information about events such as these is often needed by multiple applications within an enterprise. When a clear definition of business events or their sources does not exist, applications must deduce that business events have occurred by replicating data and applying rules. These methods cannot be done in real time and are time-consuming and error prone. The definition

> and automatic generation of business events by Service Broker can make an enterprise business event-aware. This can be done by identifying key business events based on process flows, identifying the source of each event, creating event specifications and queue architecture, modifying source applications to "fire" business events, and developing application adapters to publish events on a message bus or a specified queue. Subscriber or consumer applications can then take the events from the bus or queue and perform their own processing such as updating a database, creating a file for a batch feed, or tracking the status of an order. When key business events are available, applications can readily use them rather than having to deduce them.

Clearly, "publishing," as used in the claims of the pending application, is not equivalent to "point-to-point" communication as alleged by the *Final Office Action*, but rather is a broadcast of the relevant information to multiple subscribers. Therefore, the portion of *Suarez* cited by the *Final Office Action* showing a "point-to-point" communication is not "publishing."

b. Claims 43-78 were wrongly rejected because *Bowman-Amuah* does not teach or suggest publishing business events in accordance with the interactions between the front-office systems and the back-office systems.

The *Final Office Action* later acknowledges that *Suarez* does not explicitly disclose publishing by the enterprise integration layer, business events in accordance with the interactions between the front-office systems and back-office systems. (See, *Final Office Action*, page 6). However, the *Final Office Action* alleges that *Bowman-Amuah* discloses this feature beginning in column 76, line 21 and including the discussion of CORBA and DCOM implementations. It is not clear as to exactly what in this portion of *Bowman-Amuah* is alleged to teach or suggest publishing. However, a careful reading of *Bowman-Amuah* reveals that "publishing" is not disclosed. The Object Messaging of *Bowman-Amuah* enables objects to make requests of and receive responses from other objects through an Object Request Broker that translates the request by one object into a format appropriate for the requested object and vice versa. Thus, the Object Request Broker of *Bowman-Amuah* facilitates point-to-point communication between one object and another object by translating the messages. At most, the Object Request Broker of *Bowman-Amuah* teaches point-to-point communication between a requesting object and the Object Request Broker and another point-to-point communication between the Object Request Broker and the requested object. (See, *e.g.*, *Bowman-Amuah*, col. 76, lines 20-31). The Common Object Request Broker Architecture (CORBA) disclosed by *Bowman-Amuah* is simply a standard the provides the mechanism by which objects transparently make request and receive responses from each other, *i.e.* the mechanism for facilitating point-to-point communication

between various objects. (See, *e.g.*, *Bowman-Amuah*, col. 76, lines 43-56). The COM/DCOM section of *Bowman-Amuah* relied upon by the *Final Office Action* discloses that the Component Object Model (COM) "serves as a broker and name space keeper to connect a client and an object, but once that connection is established, the client and the object communicate directly without having the overhead of passing through a central piece of API code." (*Bowman-Amuah*, col. 77, lines 10-15). Thus, the COM of *Bowman-Amuah* facilitates point-to-point communication, but does not publish. Similarly, DCOM merely extends COM across a network. (See, *e.g.*, *Bowman-Amuah*, col. 77, lines 19-21). Clearly, neither the Object Request Broker, CORBA, COM, or DCOM disclose publishing, but merely disclose various methods for facilitating point-to-point communication between objects in a heterogeneous system.

Therefore, the *Final Office Action* has failed to establish a finding that the applied art as a whole in one form or another contains **all** of the claimed elements. Accordingly, Applicant respectfully submits that claim 43 is allowable for at least the reason that none of the applied art teach or suggest the limitation, "automatically publishing, by the enterprise integration layer, business events in accordance with the interactions between the front-office systems and back-office systems."

2. Claims 43-78 were wrongly rejected because *Suarez* in view of *Hejlsberg* and *Bowman-Amuah* does not provide any teaching or suggestion of a messaging system that automatically subscribes to the business events published by the enterprise integration layer.

Claim 43 recites, "automatically subscribing, by a messaging system coupled to the enterprise integration layer, to the business events published by the enterprise integration layer."

Assuming arguendo that the event service disclosed by *Suarez* does provide teaching of an enterprise integration layer automatically publishing business events in accordance with the interactions between the front-office systems and the back-office systems, Applicant respectfully submits that the applied art does not further provide any teaching or suggestion of the message system, as claimed.

The *Final Office Action* relied on disclosure in column 12, lines 37-64 of *Suarez* to read on these limitations. Applicant notes that this disclosure is merely directed to registering services, agents, and objects within the distributed environment. By registering, services, agents, and objects are able to be located within the distributed environment. Applicant notes that the cited portion of *Suarez* does not provide any teaching or suggestion of a messaging system or of subscribing to events published by an enterprise integration layer. Further, there is no teaching or suggestion of automatically subscribing to the events published by the event service of *Suarez*.

The *Final Office Action* attempts to cure these deficiencies by stating, "The claimed feature of 'automatically subscribes' merely automates procedures that have been well established in the area of business software, it is the examiners position that that automation of a process does not establish novelty (*In re Verner*, 120 USPQ 192, 194)." Applicant respectfully

disagrees with such an assertion. While the publish/subscribe messaging paradigm may be well established, as described in background section of the pending application in paragraph 0006, automatically subscribing to published events is not a simple automation of a previously manual process. A subscriber may not be interested in every event published and as such may subscribe to only those events that are of particular interest to the subscriber. However, the messaging system of the pending claims is interested in every event published by the enterprise integration layer, and as such may automatically subscribe to every event published by the enterprise integration layer.

As described above, the messaging system acts as another middleware layer that decouples the publishers and subscribers of events. Accordingly, the claimed messaging system makes events persistent, transforms messages for subscribers of the events (see dependent claims 50 and 51), and allows subscribers interested in an event to easily plug in to that event. Therefore, the messaging system can eliminate the need for tight integration between application programming interfaces and provide a more flexible environment.

3. Claims 43-78 were wrongly rejected because *Suarez* teaches away from the combination with *Bowman-Amuah*.

Applicant respectfully submits that *Suarez* teaches away from such a combination. As noted above, *Bowman-Amuah* discloses in column 76, lines 25-28, "An ORB enables **client objects** to access **server objects** either locally or remotely over a network and invoke operations (i.e. functions and methods) on them." Accordingly, the ORBs disclosed by *Bowman-Amuah* are used in a client-server distributed computing architecture. *Suarez* discloses many disadvantages

in distributed computer environments that use the client-server architecture in column 2, lines 36-61. Rather than using a client-server architecture for a distributed computer environment, *Suarez* discloses using agents. Applicant respectfully submits that one skilled in the art at the time of the invention would clearly recognize that these are two mutually exclusive distributed computer environment architectures.

4. Claims 43-78 were wrongly rejected because the combination of *Suarez* with *Bowman-Amuah* would change the principle of operation of *Suarez*.

Applicant respectfully submits that should ORB's replace the agents of *Suarez,* as suggested by the *Final Office Action*, the principle of operation of *Suarez* would be changed. That is, by using an ORB as the means for providing interoperability, the distributed computing architecture of *Suarez* would become a client-server architecture. If the proposed modification or combination of the prior art would change the principle of operation of the prior art invention being modified, then the teachings of the references are not sufficient to render the claims *prima facie* obvious. *In re Ratti*, 270 F.2d 810, 123 USPQ 349 (CCPA 1959)

5. Claims 43-78 were wrongly rejected because *Suarez* in view of *Hejlsberg* and *Bowman-Amuah* does not teach or suggest defining an enterprise object model which defines objects that model the data and services provided by the back-office systems.

Claim 43 recites, "defining an enterprise object model which defines objects that model the data and services provided by the back-office systems."

The *Final Office Action* relied on the disclosure in column 12, lines 37-64 to read on the limitations of the claimed "defining an enterprise object model which defines objects that model the data and services provided by the back-office systems". The cited portion of *Suarez* merely provides that information about the existence of services and agents and other objects are stored on some persistent storage. Applicant notes that there is no mention of objects that **model** the services. Rather, the cited portion of *Suarez* may be summarized with, "[T]he agents represent a standard architecture which facilitates cooperation and collaboration between agents associated with different hosts and various services" (*Suarez*: column 9, lines 28-30). Applicant respectfully submits that storage of information confirming the existence of services, agents, and objects does not **model** the services.

Applicant respectfully notes the description of enterprise object models found in paragraphs [0043]-[0044] of the pending disclosure. In contrast to the disclosure of *Suarez,* an object of the enterprise object model **models** data or services provided by the back-office systems. For example, as disclosed in paragraph [0044], an object may be mapped to/from data and services. As noted in paragraph [0044], the mappings between objects and the data or services that they model may be one-to-one, one-to-many, or many-to-many. Paragraph [0049]

discloses that a single object may be used to aggregate data from multiple back-office systems. Similarly, a single object may be broken up for storage in multiple back-office systems. Paragraph [0050] discloses that a single method call of an object may be translated into multiple service invocations in back-office systems.

Accordingly, *Suarez* does not provide any teaching or suggestion of defining an enterprise object model which defines objects that model the data and services provided by the back-office systems, as claimed.

6. Claims 43-78 were wrongly rejected because *Suarez* in view of *Hejlsberg* and *Bowman-Amuah* does not teach or suggest implementing, with a business object server of the enterprise integration layer coupled to the client access interfaces, data functions and service methods associated with the accessed objects that enable the interactions between the front-office systems and back-office systems.

Claim 43 recites, "implementing, with a business object server of the enterprise integration layer coupled to the client access interfaces, data functions and service methods associated with the accessed objects that enable the interactions between the front-office systems and back-office systems."

The *Final Office Action* relied on column 11, lines 15-43 and column 34, lines 52-67 of *Suarez* to read on these limitations. The cited portion of *Suarez* is the description of the communication between two services illustrated in Fig. 5. While the process illustrated in Fig. 5 generally does disclose an interaction between two services, the process does not teach or suggest implementing data functions and service methods associated with accessed objects of an enterprise object model, as claimed. A service is an object according to the disclosure of *Suarez*.

(See, *e.g.*, *Suarez*, col. 12, lines 37-40). If a user 13 wants to use a service, they simply access the service directly through the appropriate agent. Therefore, there is no teaching or suggestion of implementing any other data functions or service methods associated with an object prior to accessing the service.

7. Claims 43-78 were wrongly rejected because none of the limitations are optional, and conditional is not equivalent to optional.

The *Final Office Action* indicated, with regard to claim 43 and the limitation "implementing, with a business object server of the enterprise integration layer coupled to the client access interfaces, data functions and service methods associated with the accessed objects that enable the interactions between the front-office systems and back-office systems" that "Applicant(s) are reminded that optional or conditional elements do not narrow the claims because they can always be omitted." The *Final Office Action* then cites MPEP §2106 II C in support of this proposition. It is not clear whether the *Final Office Action* is asserting that this element or any element is optional and given no weight. However, this element is not, nor is any other element in the claims, an optional element as asserted by the *Final Office Action* on page 5, but is a required element. The *Final Office Action* equates "optional" with "conditional." However, this is incorrect. Optional means left to one's choice; not required or mandatory. In contrast, conditional means imposing, containing, subject to, or depending on a condition or conditions; not absolute; made or allowed on certain terms. Thus, something that is optional is not required. However, conditional does not mean that something is not required, but rather that

its occurrence is dependent upon that satisfaction of a condition. If the condition occurs, then the action depending upon the condition is required to be performed.

The *Final Office Action* relies on MPEP §2106 II C, which states "Language that suggests or makes optional but does not require steps to be performed or does not limit a claim to a particular structure does not limit the scope of a claim or claim limitation" (emphasis in original text). The case on which the MPEP relies is *In re Johnston*, 435, F.3d 1381, 77 USPQ2d 1788, 1790 (Fed. Cir. 2006) ("As a matter of linguistic precision, optional elements do not narrow the claim because they can always be omitted"). However, neither the MPEP nor *In re Johnston* include the "conditional" language relied upon by the *Final Office Action*. Furthermore, the claim language to which the Federal Circuit referred in *In re Johnston* was claim 3 of *Johnston*'s patent application which stated "further including that said wall ***may*** be smooth, corrugated, or profiled with increased dimensional proportions as pipe size is increased" (emphasis added). The Federal Circuit further stated that "[t]he board ruled that this additional content did not narrow the scope of the claim because these limitations are stated in the permissive form 'may.' As a matter of linguistic precision, optional elements do not narrow the claim because they can always be omitted. We affirm the board's ruling that claim 3 as written is anticipated."

Nothing in claim 43 expressly or impliedly makes any element of the claims optional nor does it use the permissive word "may" to describe any of the elements. Some of the elements of claim 43 may be conditional, but none are optional and as shown above, conditional is not equivalent to optional. Therefore, reliance on these references to disregard required elements of

claim 1 as optional is improper since none are optional and conditional is not equivalent to optional. Thus, all of the elements of claim 43 must be taught or suggested in the applied art.

C. 35 U.S.C. § 103(a) Rejections - Claims 56, 57, 59-61, 64, 66, 67, 69-71, and 73.

1. Claims 56, 57, 59-61, 64, 66, 67, 69-71, and 73 were wrongly rejected because claim elements were not addressed by the *Final Office Action*.

Claims 56, 57, 59-61, 64, 66, 67, 69-71, and 73 were rejected under 35 USC § 103(a) as being unpatentable over *Suarez* in view of *Hejlsberg* and further in view of *Bowman-Amuah*.

In rejecting claims 56, 57, 59-61, 64, 66, 67, 69-71, the *Final Office Action* states on page 11 that "[c]laims 52-78 are not patentably distinct from the above rejected claims and are rejected for at least the same reasons." No rationale is provided to support this assertion. Thus, the *Final Office Action* does not appear to address the limitations recited in claims 56, 57, 59-61, 64, 66, 67, 69-71, and 73 and thus fails to establish a finding that the applied art contains the elements recited in these claims. Therefore, the *Final Office Action* has failed to establish a finding that the applied art as a whole in one form or another contains **all** of the claimed elements.

As noted by the United States Supreme Court in *Graham v. John Deere Co. of Kansas City*, an obviousness determination begins with a finding that "the prior art as a whole in one form or another contains ***all***" of the elements of the claimed invention. *See Graham v. John Deere Co. of Kansas City*, 383 U.S. 1, 22 (U.S. 1966) (emphasis added).

Claims 56 and 66 include the limitation of "providing distributed transactional quality of service through a transaction processor within the enterprise integration layer." Claims 57 and 67 include the limitation of "making data persistent within a local data store of the enterprise integration layer." Claims 59 and 69 include the limitation of "using previously existing infrastructure services within the enterprise for the enterprise integration layer." Claims 60 and 70 include the limitation that "the previously existing infrastructure services are selected from a group of services consisting of a naming and directory service, a security service, and an application management and monitoring system." Claims 61 and 71 include the limitation that "the previously existing infrastructure services include each of a group of service comprising a naming and directory service, a security service, and an application management and monitoring system." Claim 64 includes the limitation of "rules regarding the transforming of the accessed common format descriptions of the data and the services into the format of the back-office systems, and rules regarding mapping each of the back-office systems to an appropriate adaptor in the set of adaptors." Claim 73 includes the limitation that "the at least one of the business events and the transformed data related to the at least one of the business events are combined in a single packet and published by the messaging interface of the messaging system or are independently published by the messaging interface of the messaging system." None of these elements are present in any of claims 43-51.

Accordingly, Applicant respectfully submits that claims 56, 57, 59-61, 64, 66, 67, 69-71, and 73 are allowable for at least the reason that the *Final Office Action* fails to establish a *prima facie* case of obviousness by failing to address the claim elements found in these claims and because none of the applied art teach or suggest the elements found in these claims.

VIII. CONCLUSION

In view of the above arguments the Appellant respectfully requests that the Final Rejection of the claims be reversed and the case advanced to issue. Should the Examiner feel that a telephone interview would advance prosecution of the present application, the Appellants invite the Examiner to call the attorneys of record.

The Commissioner is hereby authorized to charge payment of any further fees associated with any of the foregoing papers submitted herewith, or to credit any overpayment thereof, to Deposit Account No. 21-0765, of Sprint Communications Company L.P.

Respectfully submitted,
CONLEY ROSE, P.C.

Date: April 16, 2009

/Michael W. Piper/
Michael W. Piper
Reg. No. 39,800

ATTORNEY FOR APPELLANT

CONLEY ROSE, P.C.
5601 Granite Parkway, Suite 750
Plano, Texas 75024
(972) 731-2288
(972) 731-2289 (facsimile)

IX. CLAIMS APPENDIX

1-42. (Canceled)

43. (Previously Presented) A method for making computing applications throughout an enterprise aware of business events, comprising:

defining objects in an enterprise object model that model data and services provided by back-office systems;

brokering interactions, by an enterprise integration layer, between the back office systems that provide data and services and front-office systems that use the enterprise integration layer to access the data and the services provided by the back office-systems through the interactions, brokering the interactions comprising:

receiving, from the front-office systems, accesses to objects of the enterprise object model in the enterprise integration layer through client access interfaces of the enterprise integration layer, wherein each of the client access interfaces corresponds with a different technology and provides a standardized interface through which the front-office systems access the objects of the enterprise object model;

implementing, with a business object server of the enterprise integration layer coupled to the client access interfaces, data functions and service methods associated with the accessed objects that enable the interactions between the front-office systems and back-office systems; and

transforming, with a set of adapters of the enterprise integration layer coupled to the business object server, the accessed objects into a format of the back-office systems corresponding with the implementation of the data functions and the service methods associated with the accessed objects;

automatically publishing, by the enterprise integration layer, business events in accordance with the interactions between the front-office systems and back-office systems;

automatically subscribing, by a messaging system coupled to the enterprise integration layer, to the business events published by the enterprise integration layer; and

automatically generating, by the messaging system, for each of the subscribed business events a message that makes computing applications that are interested in the business event aware of the business event.

44. (Previously Presented) The method of claim 43, further comprising:

defining and storing rules in a rules engine within the enterprise integration layer, the rules including rules regarding when to automatically publish the business events in accordance with the interactions, rules regarding the transforming of the accessed objects of the enterprise object model to the format of the back-office systems, and rules regarding mapping each of the back-office systems to an appropriate adaptor in the set of adaptors,

wherein the business events are automatically published in accordance with the interactions and the rules regarding when to automatically publish the business events.

45. (Previously Presented) The method of claim 43, further comprising:

defining, in a business event repository within the enterprise integration layer, the business events that are of interest to the computing applications; and

identifying, in the business event repository, all of the publishers for each of the business events.

46. (Previously Presented) The method of claim 43, further comprising:

holding, in a metadata repository within the enterprise integration layer, metadata supplied by the set of adaptors that enables the transforming of the accessed objects of the enterprise object model to the format of the back-office systems.

47. (Previously Presented) The method of claim 43, wherein the business events are key milestones within a process flow.

48. (Previously Presented) The method of claim 43 wherein implementing data functions and service methods associated with the accessed objects further comprises:

performing one or more of object assembly, object disassembly, and service invocation functions, wherein performing object assembly includes creating a composite object

by aggregating data from a plurality of the back-office systems, performing object disassembly includes breaking a composite object into multiple objects for storage in at least one of the back-office systems, and performing service invocation includes determining which functions to invoke on one or more of the back-office systems.

49. (Previously Presented) The method of claim 43, wherein one of the business events occurs upon the implementation of the data functions and the service methods associated with the access objects, including one or more of creating data, reading data, updating data, deleting data, and invoking one of the service methods.

50. (Previously Presented) The method of claim 43, wherein the automatic generation of the message for each subscribed business event further comprises:

mapping, by one or more adaptors of a transformation layer of the messaging system, data corresponding to the business events published by the enterprise integration layer between a format of a source of the business events and a format of the computing applications.

51. (Previously Presented) The method of claim 50, further comprising:

transforming, by a source application adaptor of the one or more adaptors, data related to a business event from a format of a source of the business event to a standard data format; and

transforming, by a target application adaptor of the one or more adaptors, data from the standard data format to a format of a target subscribed to the business event.

52. (Previously Presented) A method for making computing applications throughout an enterprise aware of business events, comprising:

brokering interactions, by an enterprise integration layer, between back office systems that provide data and services and front-office systems that use the enterprise integration layer to access the data and the services provided by the back office-systems through the interactions, brokering the interactions comprising:

receiving, from the front-office systems, accesses to common format descriptions in the enterprise integration layer of the data and the services provided by the back-office systems through client access interfaces of the enterprise integration layer, wherein each of the client access interfaces correspond with a different technology and provide a standardized interface through which the front-office systems access the common format descriptions of the data and the services provided by the back-office systems;

implementing, with a business object server of the enterprise integration layer coupled to the client access interfaces, data functions and service methods associated with the accessed common format descriptions of the data and the services, wherein implementing the data functions and the service methods enable the interactions between the front-office systems and back-office systems; and

transforming, with a set of adapters of the enterprise integration layer coupled to the business object server, the accessed common format descriptions of the data and the services into a format of the back-office systems

corresponding with the implementation of the data functions and the service methods associated with the accessed common format descriptions of the data and the services;

defining and storing rules in a rules engine within the enterprise integration layer, the rules including rules regarding when to publish business events in accordance with the interactions between the front-office systems and the back-office systems;

automatically publishing, by the enterprise integration layer, the business events in accordance with the rules;

automatically subscribing, by a messaging system coupled to the enterprise integration layer, to the business events published by the enterprise integration layer; and

automatically generating, by the messaging system, for each of the subscribed business events a message that makes computing applications that are interested in the business event aware of the business event.

53. (Previously Presented) The method of claim 52, further comprising:

defining objects in an enterprise object model that model the data and the services provided by the back-office systems,

wherein the objects in the enterprise object model are the common format descriptions of the data and the services provided by the back-office systems.

54. (Previously Presented) The method of claim 52, further comprising:

defining, in a business event repository within the enterprise integration layer, the business events that are of interest to the computing applications.

55. (Previously Presented) The method of claim 52, further comprising:

holding, in a metadata repository within the enterprise integration layer, metadata supplied by the set of adaptors that enables the transforming of the accessed common format descriptions of the data and the services to the format of the back-office systems.

56. (Previously Presented) The method of claim 52, further comprising:

providing distributed transactional quality of service through a transaction processor within the enterprise integration layer.

57. (Previously Presented) The method of claim 52, further comprising:

making data persistent within a local data store of the enterprise integration layer.

58. (Previously Presented) The method of claim 52, wherein the business events are key milestones within a process flow.

59. (Previously Presented) The method of claim 52, further comprising:

using previously existing infrastructure services within the enterprise for the enterprise integration layer.

60. (Previously Presented) The method of claim 59, wherein the previously existing infrastructure services are selected from a group of services consisting of a naming and directory service, a security service, and an application management and monitoring system.

61. (Previously Presented) The method of claim 60, wherein the previously existing infrastructure services include each of a group of services comprising a naming and directory service, a security service, and an application management and monitoring system.

62. (Previously Presented) A method for making computing applications throughout an enterprise aware of business events, comprising:

brokering interactions, by an enterprise integration layer, between back office systems that provide data and services and front-office systems that use the enterprise integration layer to access the data and the services provided by the back office-systems through the interactions, brokering the interactions comprising:

receiving, from the front-office systems, accesses to common format descriptions in the enterprise integration layer of the data and the services provided by the back-office systems through client access interfaces of the enterprise integration layer, wherein each of the client access interfaces corresponds with a different technology and provides a standardized interface through which the front-office systems access the common format descriptions of the data and the services provided by the back-office systems;

implementing, with a business object server of the enterprise integration layer coupled to the client access interfaces, data functions and service methods associated with the accessed common format descriptions of the data and the services, wherein implementing the data functions and the service methods enable the interactions between the front-office systems and back-office systems; and

transforming, with a set of adapters of the enterprise integration layer coupled to the business object server, the accessed common format descriptions of the data and the services into a format of the back-office systems

corresponding with the implementation of the data functions and the service methods associated with the accessed common format descriptions of the data and the services;

defining, in a business event repository within the enterprise integration layer, business events that are of interest to computing applications in the enterprise;

identifying, in the business event repository, all publishers for each of the business events;

automatically publishing, by the enterprise integration layer, the business events in accordance with the interactions between the front-office systems and back-office systems;

automatically subscribing, by a messaging system coupled to the enterprise integration layer, to the business events published by the enterprise integration layer; and

automatically generating, by the messaging system, for each of the subscribed business events a message that makes the computing applications that are interested in the business event aware of the business event.

63. (Previously Presented) The method of claim 62, further comprising:

defining objects in an enterprise object model that model the data and the services provided by the back-office systems,

wherein the objects in the enterprise object model are the common format descriptions of the data and the services provided by the back-office systems.

64. (Previously Presented) The method of claim 62, further comprising:

defining and storing rules in a rules engine within the enterprise integration layer, the rules including rules regarding when to automatically publish the business events in accordance with the interactions, rules regarding the transforming of the accessed common format descriptions of the data and the services into the format of the back-office systems, and rules regarding mapping each of the back-office systems to an appropriate adaptor in the set of adaptors,

wherein the business events are automatically published in accordance with the interactions and the rules regarding when to automatically publish the business events.

65. (Previously Presented) The method of claim 62, further comprising:

holding, in a metadata repository within the enterprise integration layer, metadata supplied by the set of adaptors that enables the transforming of the accessed common format descriptions of the data and the services to the format of the back-office systems.

66. (Previously Presented) The method of claim 62, further comprising:

providing distributed transactional quality of service through a transaction processor within the enterprise integration layer.

67. (Previously Presented) The method of claim 62, further comprising:

making data persistent within a local data store of the enterprise integration layer.

68. (Previously Presented) The method of claim 62, wherein the business events are key milestones within a process flow.

69. (Previously Presented) The method of claim 62, further comprising:

using previously existing infrastructure services within the enterprise for the enterprise integration layer.

70. (Previously Presented) The method of claim 69, wherein the previously existing infrastructure services are selected from a group of services consisting of a naming and directory service, a security service, and an application management and monitoring system.

71. (Previously Presented) The method of claim 70, wherein the previously existing infrastructure services include each of a group of services comprising a naming and directory service, a security service, and an application management and monitoring system.

72. (Previously Presented) A method for making computing applications throughout an enterprise aware of business events, comprising:

defining objects in an enterprise object model that model data and services provided by back-office systems;

brokering interactions, by an enterprise integration layer, between the back office systems that provide data and services and front-office systems that use the enterprise integration layer to access the data and the services provided by the back office-systems through the interactions, brokering the interactions comprising:

receiving, from the front-office systems, accesses to objects of the enterprise object model in the enterprise integration layer through client access interfaces of the enterprise integration layer, wherein each of the client access interfaces corresponds with a different technology and provides a standardized interface through which the front-office systems access the objects of the enterprise object model;

implementing, with a business object server of the enterprise integration layer coupled to the client access interfaces, data functions and service methods associated with the accessed objects that enable the interactions between the front-office systems and back-office systems; and

transforming, with a set of adapters of the enterprise integration layer coupled to the business object server, the accessed objects into a format of the back-office systems corresponding with the implementation of the data functions and the service methods associated with the accessed objects;

defining and storing rules in a rules engine within the enterprise integration layer, the rules including rules regarding when to publish business events in accordance with the interactions between the front-office systems and the back-office systems;

automatically publishing, by the enterprise integration layer, the business events in accordance with the rules;

automatically subscribing, by a messaging system coupled to the enterprise integration layer, to the business events published by the enterprise integration layer;

transforming, by a source adaptor of the messaging system, data related to at least one of the business events from a format of a source of the at least one of the business events to a common data format;

publishing, by a message interface of the messaging system, the at least one of the business events and the transformed data related to the at least one of the business events in the common data format;

transforming, by a target application adaptor of the messaging system, the data related to the at least one of the business events from the common data format to a format of a target application subscribed to the at least one of the business events published by the message interface of the messaging system; and

processing the at least one of the business events by the target application.

73. (Previously Presented) The method of claim 72, wherein the at least one of the business events and the transformed data related to the at least one of the business events are combined in

a single packet and published by the messaging interface of the messaging system or are independently published by the messaging interface of the messaging system.

74. (Previously Presented) The method of claim 72, wherein the business events are key milestones within a process flow.

75. (Previously Presented) A method for making computing applications throughout an enterprise aware of business events, comprising:

defining objects in an enterprise object model that model data and services provided by back-office systems;

brokering interactions, by an enterprise integration layer, between the back office systems that provide data and services and front-office systems that use the enterprise integration layer to access the data and the services provided by the back office-systems through the interactions, brokering the interactions comprising:

receiving, from the front-office systems, accesses to objects of the enterprise object model in the enterprise integration layer through client access interfaces of the enterprise integration layer, wherein each of the client access interfaces corresponds with a different technology and provides a standardized interface through which the front-office systems access the objects of the enterprise object model;

implementing, with a business object server of the enterprise integration layer coupled to the client access interfaces, data functions and service methods associated with the accessed objects that enable the interactions between the front-office systems and back-office systems; and

transforming, with a set of adapters of the enterprise integration layer coupled to the business object server, the accessed objects into a format of the back-office systems corresponding with the implementation of the data functions and the service methods associated with the accessed objects;

defining, in a business event repository within the enterprise integration layer, business events that are of interest to computing applications in the enterprise;

identifying, in the business event repository, all publishers for each of the business events;

defining and storing rules in a rules engine within the enterprise integration layer, the rules including rules regarding when to publish the business events in accordance with the interactions between the front-office systems and the back-office systems;

automatically publishing, by the enterprise integration layer, the business events in accordance with the rules;

automatically subscribing, by a messaging system coupled to the enterprise integration layer, to the business events published by the enterprise integration layer;

automatically generating, by the messaging system, for each of the subscribed business events a message that makes the computing applications that are interested in the business event aware of the business event.

76. (Previously Presented) The method of claim 75, wherein the automatic generation of the message for each subscribed business event further comprises:

mapping, by one or more adaptors of a transformation layer of the messaging system, data corresponding to the business events published by the enterprise integration layer between a format of a source of the business events and a format of the computing applications.

77. (Previously Presented) The method of claim 76, further comprising:

transforming, by a source application adaptor of the one or more adaptors, data related to a business event from a format of a source of the business event to a standard data format; and

transforming, by a target application adaptor of the one or more adaptors, data from the standard data format to a format of a target subscribed to the business event.

78. (Previously Presented) The method of claim 75, wherein the business events are key milestones within a process flow.

X. EVIDENCE APPENDIX

None.

XI. RELATED PROCEEDINGS APPENDIX

None.